REMARKS/ARGUMENTS

Independent claims 1, 11, 14 and 15 have been amended to recite an outputting step to more clearly define the practical application of the claims.

Applicant has also amended claims 6, 13 and 15 in response to the Examiner's interpretation of the term "ELSE" to mean "OR". Applicant has amended claims 6, 13 and 15 by replacing the if, else if and else statements with conditional upon determining steps. Applicant submits that these conditional upon determining steps cannot be construed to mean "OR".

Applicant has also updated the Summary of the Invention section of the specification to correspond to the amended claims.

Information Disclosure Statement

In paragraph 1 of the detailed action, the Examiner has stated that the Information Disclosure Statement (IDS) that was submitted in September of 2003 has not been considered because the title, author and number of pages are not provided for each of the documents listed.

In response, Applicant submits a replacement Form PTO/SB/08B, which includes the title, author and number of pages of each of the documents listed in the IDS that was submitted in September of 2003, as required.

Specification

In paragraph 3 of the detailed action, the Examiner has stated that a new title is required that is clearly indicative of the invention to which the claims are directed and that legal words like "method" and "apparatus" should be removed.

In response, Applicant respectfully submits that the present title, namely "A METHOD, SYSTEM AND SOFTWARE FOR TRANSPARENT INTERFACE MIGRATION", is clearly indicative of the invention to which the claims are directed, as the claims recite a method (claims 1-10), a system (claims 11-13), and software stored on a computer readable medium (claims 14 and 15), for transparent interface migration. Furthermore, Applicant respectfully submits that there is no reasons that words like "method" and "apparatus" should not be used in

the title of an invention. The USPTO has a long history of allowing patents that include precisely these words in their titles.

Claim Rejections – 35 U.S.C. 101

In paragraph 4 of the detailed action, the Examiner has rejected claims 1-15 under 35 U.S.C. 101 on the ground that the claimed invention is directed to non-statutory subject matter. The Examiner has stated that the current focus of the Patent Office in regard to statutory invention under 35 U.S.C. 101 for method claims and claims that recite judicial exceptions (software) is that the claimed invention recite a practical application, which can be provided by a physical transformation or a useful, concrete and tangible result. The Examiner has argued that the final result of the claims is an interface wrapper which is not a tangible result because the result are not written or updated or displayed on a computer readable medium, which would make the claims tangible.

In response, Applicant submits that the amended independent claims 1, 11, 14 and 15 recite the tangible step of outputting the interface wrapper, which allows a software application to communicate with a second interface rather than a first interface. Applicant submits that outputting the interface wrapper is a concrete and tangible result and therefore the subject matter of the independent claims and the dependent claims which depend therefrom are clearly statutory subject matter under 35 U.S.C. 101.

Claim Objections

In paragraph 5 of the detailed action, the Examiner has objected to claims 5, 6 and 12-14 on the ground that these claims contain more then one period.

In response, Applicant has replaced the periods in claims 5, 6 and 12-14, which were used to indicate the instantiation of method steps, with hyphens. For example, the method step of claim 5 that was previously labelled as a.1) is labelled as a-1) in amended claim 5.

Claim Rejections – 35 U.S.C. 102

In paragraph 7 of the detailed action, the Examiner has rejected claims 1-6 and 10-15 under 35 U.S.C. 102 (b) as being anticipated by the cited Template Software. The Examiner has referred to three template software documentation documents in support of the objections of claims 1-6 and 10-15 under 35 U.S.C. 102 (b), namely: using the SNAP communication component (referred to as COM); using the SNAP development environment (referred to as SNAP); and using the SNAP permanent storage component (referred to as PERM).

In response, before setting forth a discussion of the cited art applied in the Office Action, it is respectfully submitted that controlling case law has frequently addressed rejections under 35 U.S.C. 102. "For a prior art reference to anticipate in terms of 35 U.S.C. Section 102 every element of the claimed invention must be identically shown in a single reference." *Diversitech Corp. v. Century Step, Inc.*, 850F. 2d 675, 677, 7 U.S.P.Q. 2d 1315, 1317 (Federal Circuit 1988). "If any claim, element, or step is absent from the reference that is being relied upon, there is no anticipation." *Closter Speedsteel AB v. Crucible, Inc.*, 793F. 2d 1565, 230 U.S.P.Q. 81 (Federal Circuit 1986). The following analysis of the present rejections is respectfully offered with guidance from the foregoing controlling case law decisions.

As will be discussed in greater detail below, key features of the subject claims which are absent from the cited art are: creating a computer readable mapping between a first interface and a second interface; and an auto-generator, which uses the computer readable mapping to automatically generate an interface wrapper to replace the first interface and allow a software application to use the second interface rather than the first interface.

To begin, the main piece of the cited template software is the SNAP development environment, which provides a development environment for object oriented applications. The portions of the SNAP development environment which the Examiner has argued are relevant to the present invention are the SIB connection editor, the database mapping editor and the schema editor. The SIB connection editor is used to edit a shared information based (SIB) connection, which is the mechanism that enables two disparate processes to communicate using TCP/IP protocols (see SNAP page 5-3). In order for the two processes to communicate they must contain classes that have the same class name, contain the same set of attributes and are defined

as shared. The SIB connection editor can then be used to define which shared classes are being passed over a connection between the two processes. If the shared classes in both of the processes are identical, then the sharing is very straight forward. However, if the shared classes are not identical, i.e. if the shared class in one of the processes contains attributes that are not present in the identically named shared class in the other process, then the attributes that are not common to the two shared classes must be sub-classed in the shared class in the process in which they are present, leaving only the attributes that are common to the two shared classes in the parent class of the sub-class (see COM page 5-5). In this case, the sub-class is referred to as the mapped sub-class and the parent of the mapped sub-class is referred to as the mapped class. The SIB connection editor can be used to point the attributes of the mapped class to the attributes of the mapped sub-class such that although the sharing of attributes happens between the shared class of one process and the mapped class of the second process, the shared attributes are ultimately stored in the attributes of the mapped sub-class. It should be pointed out that although the SIB connection editor can specify the mapping between the attributes of a mapped class and a mapped sub-class, the SIB connection editor cannot create a shared class nor can it create a mapped sub-class within a mapped class in order to allow the mapped class to communicate with a shared class with the same attributes as the mapped class.

While the SIB connection editor and the SIB connection allow for attribute sharing between the shared classes of two disparate processes, the database mapping editor and database connections allow processes to access databases by mapping the data tables of a database of a database management system (DBMS) to the class attributes of a process. A database mapping specifies which SNAP class attributes correspond to which database table columns. In order to create a database mapping, a user must designate a database connection, a SNAP class, and one or more database tables. The database connection specifies the DMBS used to access the database and the database to be mapped. The user then maps the attributes of the SNAP class to the corresponding database table columns (see SNAP pages 6-3 to 6-4). It is important to note that no SIB connection is involved in a database mapping or database connection.

Schemas are used in the template software to traverse an object model and operate on specified objects and attributes of interest. The user can specify schemas when the user calls certain predefined SNAP functions. The objects and attributes designated by the schema are then

operated on by these functions. The user creates a schema for a class and defines the schema to designate a subset of the class's attributes. If the user defines a schema that designates a relation attribute, i.e. an attribute of the class that points to the attribute of another class, the user can also specify another schema that describes how to traverse the object of the class referred to by that relation attribute. In this way, the schema allows an application to operate on all of the attributes designated in the schema with a single function call rather than looping through each objects to operate on the attributes individually (see COM page 12-2). The schema editor allows a user to specify a schema by first creating an object for each of the attributes that are to be operated on by the schema in a SCHEMA ATTR class, which defines whether or not the attribute is a relation attribute as well as the name of the attribute and the class that contains the attribute. The schema editor specifies which attributes are to be included in the schema by creating an object in a SCHEMA class, which specifies objects of the SCHEMA ATTR class that represents attributes that will be designated by the schema as well as the class which the schema describes. In this way, the SCHEMA ATTR objects point to individual attributes and the SCHEMA object identifies which of these attributes are included in the schema. It is important to note that the template software documentation specifically states that only instance attributes can be set or retrieved by a schema, i.e. classes, objects, static attributes and methods cannot be transferred by a schema (see COM page 4-26).

From the foregoing, it is clear that the features of the template software, namely the SIB connections created by the SIB connection editor, the database connections created by the database mapping editor and the schema created by the schema editor, are solely directed to the sharing of attributes between one class and another in the case of SIB connections and schemas and between a class and a database in the case of the database connection.

The present invention is directed to facilitating the migration of a software application from a first interface to a second interface by: creating a computer readable mapping between the first interface and the second interface; running the mapping through an autogenerator which uses the mapping to automatically generate and output an interface wrapper, which replaces the first interface and is interposed between the second interface and the software application in order to allow the software application to communicate with the second interface.

With regard to claim 1, the Examiner has alleged that the method of migrating a software application to allow the software application to use a second interface instead of a first interface recited in claim 1 is disclosed on PERM pages 3-28 to 3-29, which allegedly discloses "changing the SIB connection and leaving the mapping in place-e.g. change from ODBC SIB connection to Oracle". However, as indicated by the Examiner, pages 3-28 and 3-29 of PERM relate to database connections and database mapping, which as indicated above do not include a SIB connection. Furthermore, it is not clear which type of mapping the Examiner is referring to in this case, as the Examiner has referred to both database mapping and schema mapping in support of the rejection of claim 1. It is assumed that the Examiner is referring to database mapping, as the Examiner's example of changing from ODBC SIB connection to Oracle is a change between database connections, although, as indicated above, a change between database connections would not include the change of a SIB connection.

In addition, claim 1 recites "creating a computer readable mapping between the first interface and the second interface", which the Examiner has alleged is disclosed by the SIB connection of the template software. However, as discussed above, a SIB connection is merely the mechanism by which two classes may share their common attributes, i.e. a SIB connection is the mapping of a shared attribute of a local class with a shared attribute of a first remote class. If the local class is to share an attribute with a second remote class, a second SIB connection must be created to map the shared attributes of the local class to the shared attributes of the second remote class. If it is assumed that the local class is part of a software application and the first remote class is part of a first interface, while the second remote class is part of a second interface, it is clear from the foregoing that a first SIB connection is required for communication between the software application (i.e. the local class) and the first interface (i.e. the first remote class), while a second SIB connection is required for communication between the software application (i.e. the local class) and the second interface (i.e. the second remote class). Therefore, creating a SIB connection in the template software is not the same as creating a computer readable mapping between the first interface and the second interface as recited in claim 1. Furthermore, the Examiner has alleged that "running the mapping through an autogenerator wherein the autogenerator uses the mapping to automatically generate an interface wrapper", as recited in claim 1, is disclosed by the SIB editor of the template software. However, as discussed above, a

SIB connection is not a computer readable mapping between the first interface and the second interface and the SIB editor merely creates SIB connections, it does not operate on a computer readable mapping between a first interface and a second interface to automatically generate an interface wrapper as recited in claim 1.

In view of the foregoing, Applicant respectfully submits that the template software fails to teach or fairly suggest key limitations of claim 1 and therefore claim 1 distinguishes over the teachings of the template software and its associated support documentation.

Independent claims 11, 14 and 15 also recite the distinguishing features of claim 1 and therefore distinguish over the template software and its associated support documentation for at least the same reasons as discussed above with respect to claim 1. In addition, independent claims 14 and 15 recite additional technical features which distinguish over the template software and its associated support documentation. For example, claims 14 and 15 recite adding "code from the computer readable mapping related to the method to the class to be included in the interface wrapper", whereas, as discussed above, the SIB connections, schema and database connections of the template software are solely directed to the transfer of attributes of a class and are silent as to the methods of a class.

Dependent claims 2-10 which depend from claim 1, and dependent claims 12 and 13 which depend from claim 11, distinguish over the teachings of the template software and its associated support documentation for at least the same reasons as discussed above with respect to independent claims 1 and 11, respectively. Dependent claims 2-10 and 12-13 further distinguish over the template software and its associated support documentation by reciting additional technical features which are not taught by the template software and its associated support documentation. For example, claims 5, 6, 12 and 13 recite adding "code from the computer readable mapping related to the method to the class to be included in the interface wrapper", which distinguishes over the template software and its associated support documentation which is solely directed to the sharing of class attributes and is silent on class methods as discussed above.

In view of the fact that the template software and its associated support documentation fail to teach key limitations of the claims, and also fail to teach every element of the claimed

invention, as is required under 35 U.S.C. 102, given the ruling in *Closter Speedsteel AB v. Crucible, Inc.* and *Diversitech Corp. v. Century Step, Inc.*, the Examiner is respectfully requested to withdraw the 35 U.S.C. 102(b) rejection of claims 1-6, 10-15.

Claim Rejection 35 U.S.C. 103

In paragraph 9 of the detailed action, the Examiner has rejected claims 7-9 under 35 U.S.C. 103(a) as being unpatentable over the template software in view of Enterprise Application Integration with XML and JAVA, by J.P. Morgenthal *et al.*, 2001, Chapter 5 (XML).

To begin, Applicant respectfully submits that a first criterion required to establish *prima facie* obviousness has not been satisfied. That is, the cited template software and XML do not teach all of the claimed features.

Claims 7-9 are dependent on claim 1. As outlined above in response to the 35 U.S.C. 102(b) rejections, the template software and its associated support documentation fail to teach key limitations of claim 1. As the XML also fails to teach these key limitations of claim 1, claim 1 distinguishes over both the template software and the XML reference and all claims that depend from claim 1, including claims 7-9, distinguish over the template software and the XML reference for at least the same reasons.

Since the template software and its associated support documentation and the XML reference failed to teach key limitations of the present invention, the first criteria for the *prima facie* case of obviousness has not been satisfied. Applicant therefore respectfully submits that claims 7-9 are patentable over the template software and its associated support documentation and the XML reference since a case of *prima facie* obviousness cannot be established.

In view of the foregoing, early favorable consideration of this application is earnestly solicited. In the event that the Examiner has concerns regarding the present response, the Examiner is encouraged to contact the undersigned at the telephone number listed below.

Respectfully submitted,

MARTIN SOUKUP

By _____
Philip Lapin
Reg. No. 44,443
Tel.: (613) 232-2486

Date: November 30, 2006

PDL:JFS:mcg
Encl.